I CLAIM:

1. A first-in, first-out memory device comprising:

a memory array having a plurality of address locations for storing incoming data;

a boundary pointer for indicating an end point of a buffer area formed within said memory array into which said incoming data can be stored; and

a controller for adjusting the value of said boundary pointer in accordance with the amount of incoming data to be stored.

2. The first-in, first-out memory of claim 1 further comprising:

a read pointer, coupled to said memory array, for indicating a read address of said buffer area; and

a write pointer, coupled to said memory area, for indicating a write address of said buffer area.

3. The first-in, first-out memory of claim 1, wherein said memory array is implemented utilizing a ring buffer.

4. The first-in, first-out memory of claim 3, wherein said controller operates to move said boundary pointer so as to increase the size of said buffer on the basis of a 1:1 correspondence with the amount of incoming data.

5. The first-in, first-out memory of claim 1, wherein said controller dynamically varies the value of said boundary pointer during operation in response to the amount of said incoming data to be stored.

6. A first-in, first-out memory device comprising:

a memory array having a plurality of address locations for storing incoming data;

a first boundary pointer for indicating an end point of a first buffer area formed within said memory array into which said incoming data can be stored;

a second boundary pointer for indicating an end point of a second buffer area formed within said memory array into which said incoming data can be stored; and

a controller for adjusting the value of said first boundary pointer and said second boundary pointer in accordance with the amount of incoming data to be stored.

7. The first-in, first-out memory of claim 6 further comprising:

a first read pointer, coupled to said memory array, for indicating a read address of said first buffer area;

a first write pointer, coupled to said memory area, for indicating a write address of said first buffer area

a second read pointer, coupled to said memory array, for indicating a read address of said second buffer area; and

a second write pointer, coupled to said memory area, for indicating a write address of said second buffer area.

8. The first-in, first-out memory of claim 6, wherein said memory array is implemented utilizing a ring buffer.

9. The first-in, first-out memory of claim 8, wherein said controller operates to move said first boundary pointer and said second boundary pointer so as to increase the size of said first buffer and said second buffer on the basis of a 1:1 correspondence with the amount of incoming data.

10. The first-in, first-out memory of claim 6, wherein said controller dynamically varies the value of said first boundary pointer and said second boundary pointer during operation in response to the amount of said incoming data to be stored.

11. A method of storing data in a first-in, first-out memory device, said method comprising the steps of:

providing a memory array having a plurality of address locations for storing incoming data;

defining a boundary pointer for indicating an end point of a buffer area formed within said memory array into which said incoming data can be stored; and

adjusting the value of said boundary pointer in accordance with the amount of incoming data to be stored.

12. The method of storing data in a first-in, first-out memory of claim 11, further comprising the step of:

defining a read pointer for indicating a read address of said buffer area; and

defining a write pointer for indicating a write address of said buffer area.

13. The method of storing data in a first-in, first-out memory of claim 11, wherein said memory array is implemented utilizing a ring buffer.

14. The method of storing data in a first-in, first-out memory of claim 13, wherein said boundary pointer is modified so as to increase the size of said buffer on the basis of a 1:1 correspondence with the amount of incoming data.

15. The method of storing data in a first-in, first-out memory of claim 11, wherein the value of said boundary pointer is dynamically varied during operation in response to the amount of said incoming data to be stored.

16. A method of storing data in a first-in, first-out memory device, said method comprising the steps of:

defining a memory array having a plurality of address locations for storing incoming data;

defining a first boundary pointer for indicating an end point of a first buffer area formed within said memory array into which said incoming data can be stored;

defining a second boundary pointer for indicating an end point of a second buffer area formed within said memory array into which said incoming data can be stored; and

adjusting the value of said first boundary pointer and said second boundary pointer in accordance with the amount of incoming data to be stored.

17. The method of storing data in a first-in, first-out memory of claim 16 further comprising the step of:

defining a first read pointer for indicating a read address of said first buffer area;

defining a first write pointer for indicating a write address of said first buffer area

defining a second read pointer for indicating a read address of said second buffer area; and

defining a second write pointer for indicating a write address of said second buffer area.

18. The method of storing data in a first-in, first-out memory of claim 16, wherein said memory array is implemented utilizing a ring buffer.

19. The method of storing data in a first-in, first-out memory of claim 18, wherein said first boundary pointer and said second boundary pointer are adjusted so as to increase

the size of said first buffer and said second buffer on the basis of a 1:1 correspondence with the amount of incoming data.

20. The method of storing data in a first-in, first-out memory of claim 16, wherein said first boundary pointer and said second boundary pointer are dynamically varied during operation in response to the amount of said incoming data to be stored.